Enumeration Kernels for Vertex Cover and Feedback Vertex Set

Marin Bougeret
LIRMM, Université de Montpellier, Montpellier, France

Guilherme C. M. Gomes
LIRMM, Université de Montpellier, CNRS, Montpellier, France
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Vinicius F. dos Santos
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Ignasi Sau
LIRMM, Université de Montpellier, CNRS, Montpellier, France

Abstract

Enumerative kernelization is a recent and promising area sitting at the intersection of parameterized complexity and enumeration algorithms. Its study began with the paper of Creignou et al. [Theory Comput. Syst., 2017], and development in the area has started to accelerate with the work of Golovach et al. [J. Comput. Syst. Sci., 2022]. The latter introduced polynomial-delay enumeration kernels and applied them in the study of structural parameterizations of the Matching Cut problem and some variants. Few other results, mostly on Longest Path and some generalizations of Matching Cut, have also been developed. However, little success has been seen in enumeration versions of Vertex Cover and Feedback Vertex Set, some of the most studied problems in kernelization. In this paper, we address this shortcoming. Our first result is a polynomial-delay enumeration kernel with $2k$ vertices for Enum Vertex Cover, where we wish to list all solutions with at most k vertices. This is obtained by developing a non-trivial lifting algorithm for the classical crown decomposition reduction rule, and directly improves upon the kernel with $\mathcal{O}(k^2)$ vertices derived from the work of Creignou et al. Our other result is a polynomial-delay enumeration kernel with $\mathcal{O}(k^3)$ vertices and edges for Enum Feedback Vertex Set; the proof is inspired by some ideas of Thomassé [TALG, 2010], but with a weaker bound on the kernel size due to difficulties in applying the q-expansion technique.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Kernelization, Enumeration, Vertex cover, Crown decomposition, Feedback vertex set

Digital Object Identifier 10.4230/LIPIcs.IPEC.2025.23

Related Version *Full Version*: https://arxiv.org/abs/2509.08475

Funding *Guilherme C. M. Gomes*: Funded by the European Union, project PACKENUM, grant number 101109317. Views and opinions expressed are however those of the author only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.
Vinicius F. dos Santos: Supported by CNPq (Grants 312069/2021-9, 406036/2021-7, 404479/2023-5, and 308129/2025-3).
Ignasi Sau: French project ELIT (ANR-20-CE48-0008-01).

20th International Symposium on Parameterized and Exact Computation (IPEC 2025).
Editors: Akanksha Agrawal and Erik Jan van Leeuwen; Article No. 23; pp. 23:1–23:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Enumeration problems are central objects in both practical and theoretical computer science, with the main example of the former being the branch-and-bound method [35] (and its relatives), universally used in integer programming solvers. Instead of answering a question in either the positive or the negative, in this class of problems the goal is to generate all witnesses, or *solutions*, that satisfy the given problem's constraints, or decide that no solution exists; we denote by $\mathsf{Sol}(x)$ the solution set of an instance x of some fixed problem of interest. Typically, the number of solutions will be exponential in the input size n, and thus *input-sensitive* algorithms, whose running-times are only analyzed with respect to n, are not powerful enough tools to capture all nuances intrinsic to enumeration problems. In particular, *input-polynomial* algorithms, whose running times are of the form $n^{\mathcal{O}(1)}$, are inadequate models of efficient enumeration. Consequently, enumeration algorithms are usually analyzed in the *output-sensitive* context, i.e., their running times depend on both n and the size of the output. In this setting, a popular class of efficiently solvable enumeration problems are those that admit *polynomial-delay* [3, 23, 31] algorithms, where the running time between consecutive outputs must be bounded by $\mathsf{poly}(n)$. These algorithms are unfeasible goals in many interesting cases, as enumeration problems for which the existence of a single solution is an NP-hard problem trivially do not admit such algorithms unless P = NP.

Orthogonally to enumeration, the theory of parameterized complexity [18, 22, 28] gained significant traction as a means of coping with NP-hardness. In this framework, an instance x is equipped with a *parameter* of interest $k = \kappa(x)$ and the efficient algorithms, known as *fixed-parameter tractable* (FPT), are those of running time $f(k) \cdot n^{\mathcal{O}(1)}$, for some computable function f. An equivalent definition of tractability in this framework, and central to our work, is that of *decision kernelization*: a polynomial-time algorithm, or *kernel*, that compresses an instance (x, k) into an equivalent instance (y, ℓ) such that $|y|, \ell \leq g(k)$ for some computable function g, known as the *size* of the kernel. Beyond their theoretical interest, kernels are also models for theoretically guaranteed *pre-processing* algorithms, a key feature in real-world applications, such as in integer optimization. Polynomial kernels, i.e., when $g(k) = \mathsf{poly}(k)$, are of particular interest, as they typically result in significant reductions in input size. Kernelization theory counts with several robust results and tools [1, 4, 6, 8, 11, 21, 25, 28, 39], including meta-theorems [7, 27, 29, 32], for both upper and lower bounds.

A very successful case study in kernelization is that of Vertex Cover, arguably the most fundamental problem in parameterized complexity. Abu-Khzam et al. [1] presented the first kernel with $\mathcal{O}(k)$ vertices for the problem when parameterized by the solution size k, using the then-novel concept of *crown decompositions* to improve on the previously best known kernel with $\mathcal{O}(k^2)$ vertices, implied by a result of Buss and Goldsmith [13]. Currently, the best known kernels for Vertex Cover have size $2k$ [15], and are based on the half-integral relaxation of Nemhauser and Trotter [38]. Alongside Vertex Cover, the Feedback Vertex Set problem has been quite influential in kernelization theory [9, 12, 39]. When parameterized by the solution size k, it was first shown to admit a kernel with $\mathcal{O}(k^{11})$ vertices by Burrage et al. [12]; this was later improved to a cubic kernel by Bodlaender and van Dijk [9]. Shortly afterwards, Thomassé [39] presented a kernel with $\mathcal{O}(k^2)$ vertices and edges by introducing a generalization of crown decompositions known as *q-expansions*, which have since been used in other examples in the literature [25, 26, 33, 34]. Thomassé's kernel size has also been shown to be asymptotically optimal by Dell and Marx [21].

Over the last years, there have been mounting efforts to extend parameterized complexity to the enumeration setting [5, 14, 17, 19, 20, 24, 30, 33, 36, 37], with *input*-FPT and FPT-*delay* having their expected definitions; the work of Creignou et al. [17] has been particularly influential in this endeavor, and we refer to it for a broader discussion on parameterized enumeration. We will also not further discuss input-sensitive enumeration, focusing on delay-based algorithms. While parameterized enumeration algorithms have been developed for some decades, it took much longer, however, for the area of *enumerative kernelization* to emerge, even though kernelization itself did play a significant role in several works [17, 19, 24].

Only very recently Golovach et al. [30] defined *polynomial-delay enumeration (PDE) kernels*, which are equivalent to the existence of FPT-delay algorithms, much like FPT algorithms and kernels are equivalent in the decision world; previous models, such as full kernels [19] and enum-kernels [17], had issues that made them unsatisfactory definitions for enumerative kernelization. Intuitively, a PDE kernel has two parts: the first, $\mathcal{A}_1$, is the *compression algorithm* and has the same constraints as a decision kernel, while the second $\mathcal{A}_2$, known as the *lifting algorithm*, receives the input (x, k), the output (y, ℓ) of $\mathcal{A}_1$, and a solution $Y \in \mathsf{Sol}(y)$, and must output, with $\mathsf{poly}(|x| + |y| + k + \ell)$-delay, a non-empty subset $S_Y \subseteq \mathsf{Sol}(x)$; additionally, the S_Y's must partition $\mathsf{Sol}(x)$. PDE kernels have been applied to some problems [14, 30, 33, 36], mainly restricted to variants of the Matching Cut problem; [33] is the unique exception, as it investigates an enumeration variant of the Longest Path problem; we remark that they apply the q-expansion technique to obtain their results. Despite being stated as an enum-kernel, Creignou et al.'s [17] $\mathcal{O}(k^2)$ kernel for Enum Vertex Cover is a PDE kernel. Despite these positive results, the most basic techniques of decision kernelization theory seem quite difficult to adapt to the enumeration setting. Motivated by this phenomenon, we examine the best known kernels for Vertex Cover and Feedback Vertex Set and seek to adapt their ideas and techniques to their respective enumeration variants.

Our contributions. Our first result is a PDE kernel for Enum Vertex Cover with $2k$ vertices, and directly improves upon the $\mathcal{O}(k^2)$ kernel of Creignou et al. [17]; the problem is formally defined as follows.

Enum Vertex Cover
Instance: A graph G and an integer k.
Enumerate: All vertex covers of G of size at most k.

Our proof is a direct generalization of the kernel with $2k$ vertices for Vertex Cover; this, in turn, is based on the celebrated crown decomposition technique. Specifically, we obtain the following theorem.

▶ **Theorem 1.** Enum Vertex Cover *admits a PDE kernel with at most* $2k$ *vertices when parameterized by the maximum desired size of a solution* k*.*

Intuitively, our kernel works by applying the classic crown reduction that deletes the head H and crown C of the decomposition, reduces k by $|H|$, and returns only the body B. The $2k$ bound is consequence of the Nemhauser-Trotter decomposition theorem [38] and the fact that it either yields a crown decomposition or gives a bound on the size of the input graph. The complicated part is showing how to design a lifting algorithm; we do so by proving that Enum Vertex Cover restricted to *crowned graphs*, i.e., graphs that can be partitioned into $H \cup C$ where C is an independent set and there is an H-saturating matching between H and C, admits a polynomial-delay algorithm. Given a solution S of the reduced instance, our lifting algorithm has three phases: (i) brute force how the *slack* $k - (|S| + |N(B \setminus S) \cap H|)$ is distributed in C, (ii) compute sets of vertices forced to be added to and forbidden in the solution given the output of (i), and (iii) solve the instance $(G', |V(G')|/2)$ induced by the remaining vertices. The key observations are that: (a) every configuration tested in (i) leads to a solution of the input, and (b) G' is a crowned graph with a perfect matching.

We then proceed to show a cubic PDE kernel for the Enum Feedback Vertex Set problem, formally defined as below.

Enum Feedback Vertex Set
Instance: A multigraph G and an integer k.
Enumerate: Every feedback vertex set of G of size at most k.

We follow a similar direction to Thomassé's quadratic kernel [39]: we first design reduction rules to lower bound the minimum degree of the instance; then, we show that upper bounding the maximum degree is enough to lead to a kernel; and finally, we present reduction rules that yield said upper bound. The core ingredient of Thomassé's proof – the computation of a 2-expansion in an auxiliary graph – seems to fail in the enumeration setting, and we are unable to generalize it. Nevertheless, we obtain Theorem 2. Our set of reduction rules is mostly different, which leads to a quadratic instead of a linear bound on the maximum degree. As our lower bound on the minimum degree is also different, we must prove a slight generalization on the bound used in Thomassé's proof.

▶ **Theorem 2.** *There is a PDE kernel for* Enum Feedback Vertex Set *parameterized by the size of the solution k with $\mathcal{O}(k^3)$ vertices.*

Organization. In Section 2 we present some basic preliminaries about graphs, parameterized complexity, and enumeration problems. In Section 3, we present our linear PDE kernel for Enum Vertex Cover. In Section 4, we present our cubic PDE kernel for Enum Feedback Vertex Set and discuss some challenges in using rules based on q-expansions. We present concluding remarks and possible research directions in Section 5. Due to space limitations, the proofs of the statements marked with "($\star$)" have been omitted and can be found in the full version available online (`https://arxiv.org/abs/2509.08475`).

2 Preliminaries

We denote the set $\{1, 2, \ldots, n\}$ by $[n]$. We use standard graph-theoretic notation, and we consider simple undirected graphs without loops or multiple edges; see [10] for any undefined terminology. When the graph is clear from the context, the degree (that is, the number of neighbors) of a vertex v is denoted by $\deg(v)$, and the number of neighbors of a vertex v in a set $A \subseteq V(G)$ and its neighborhood in it are, respectively, denoted by $\deg_A(v)$ and $N_A(v)$; we also define $N(S) = \bigcup_{v \in S} N(v) \setminus S$. A *matching* M of a graph G is a subset of edges of G such that no vertex of G is incident to more than one edge in M; for simplicity, we define $V(M) = \bigcup_{uv \in M} \{u, v\}$ and refer to it as the set of *M-saturated vertices*. The *subgraph of G induced by X* is defined as $G[X] = (X, \{uv \in E(G) \mid u, v \in X\})$. A *feedback vertex set* $X \subseteq V(G)$ is such that $G \setminus X \coloneqq G[V(G) \setminus X]$ is a forest; the *feedback vertex number* of G is the size of a feedback vertex set of minimum size. Given adjacent $u, v \in V(G)$, a *contraction of u into v* is an operation that outputs the graph $G \setminus \{u\}$ with the added edges $\{vw \mid w \in N_G(u) \setminus \{v\}\}$. An algorithm is said to run with *polynomial delay* [16] if the times before the first output (called the *precalculation period*), after the last output (*the postcalculation period)*, and in between two consecutive outputs are bounded by a polynomial in the input size only.

We refer the reader to [18, 22] for basic background on parameterized complexity and present here only definitions pertaining to parameterized enumeration.

▶ **Definition 3** (Parameterized enumeration problem (Creignou et al. [17], Golovach et al. [30])). *A* parameterized enumeration problem *over a finite alphabet* Σ *is a tuple* $\Pi = (L, \mathsf{Sol}, \kappa)$*, shorthanded by* Π_κ*, such that:*

i. $L \subseteq \Sigma^\star$ *is a decidable language;*
ii. $\mathsf{Sol} : \Sigma^\star \mapsto 2^{\Sigma^\star}$ *is a computable function such that, for every* instance $x \in \Sigma^\star$, $\mathsf{Sol}(x)$ *is finite and we have* $\mathsf{Sol}(x) \neq \emptyset$ *if and only if* $x \in L$*;*
iii. $\kappa : \Sigma^\star \mapsto \mathbb{N}$ *is the* parameterization*.*

An instance x *of* Π_κ *is a* no-instance *if* $\mathsf{Sol}(x) = \emptyset$*, and is a* yes-instance *otherwise.*

▶ **Definition 4** (Polynomial-delay enumeration kernel). *Let* Π_κ *be a parameterized enumeration problem. A* polynomial-delay enumeration kernel*,* PDE kernel *for short, is a pair of algorithms* $\mathcal{A}_1, \mathcal{A}_2$ *such that, given an instance* x *of* Π_κ *and* $\kappa(x)$*:*

- *The* compression algorithm $\mathcal{A}_1$ *runs in* $\mathsf{poly}(|x| + \kappa(x))$*-time and outputs an instance* y *of* Π_κ *satisfying* $\kappa(y), |y| \leq f(\kappa(x))$*, with* f *being a computable function, and with* $\mathsf{Sol}(y) \neq \emptyset$ *if and only if* $\mathsf{Sol}(x) \neq \emptyset$*;*
- *The* lifting algorithm $\mathcal{A}_2$ *receives* x*,* $y = \mathcal{A}_1(x, \kappa(x))$*,* $\kappa(x)$*,* $\kappa(y)$*, some* $Y \in \mathsf{Sol}(y)$*, and outputs, with polynomial delay on its input, a non-empty* $S_Y \subseteq \mathsf{Sol}(x)$*. Moreover,* $\{S_Y \mid Y \in \mathsf{Sol}(y)\}$ *is a partition of* $\mathsf{Sol}(x)$*.*

A *graph subset problem* is a problem with a graph as input and a solution being a subset of its vertices. We say that a PDE kernel for a graph subset problem is *extension-only* if, given a solution $Y \in \mathsf{Sol}(y)$, the lifting algorithm works by only adding some vertices in $x \setminus y$ to Y. Intuitively, this constraint implies that a solution of the input instance x cannot be generated by two different solutions of the compressed instance y, so it only remains to prove that every solution of x is generated from some solution of y and every solution of y leads to some solution of x. Both kernels developed in this work are extension-only. We remark that the kernels for Enum Matching Cut and its variants developed by Golovach et al. [30] are not extension only, as their lifting algorithms remove some vertices of the compressed instance's solutions.

3 A kernel with $2k$ vertices for Enum Vertex Cover

In this section we prove Theorem 1. Our main tools for this result are crowned graphs and crown decompositions, given in Definition 5.

▶ **Definition 5.** *A graph is a* crowned graph *if its vertex set can be partitioned into* $H \cup C$*, such that* C *is an independent set, and such that there is a matching* M *between* H *and* C *where* $|M| = |H|$*. The set* C *is called the* crown*, and* H *is called the* head*. Given a graph* G*, a* crown decomposition *[18] of width* t *is a partition* (C, H, B) *of* $V(G)$ *such that* $G[H \cup C]$ *is a crowned graph (with head* H*),* $N_G(C) = H$*, and* $|H| = t$*. Set* B *is called the* body*.*

To that end, we first reduce the task of finding a PDE kernel to an enumeration problem Enum Crown, and then reduce Enum Crown to an even more structured enumeration problem Enum Small Crown for which we provide a polynomial-delay enumeration algorithm. Let us first define our problem.

ENUM CROWN **Instance:** A crowned graph $G = (H \cup C, E)$ and an integer x in $[0, |C|]$. **Enumerate:** All vertex covers of G of size exactly $|H| + x$.

The problem ENUM SMALL CROWN corresponds to the special case of ENUM CROWN where $|C| = |H|$ and $x = 0$. Given an input (G, x) of ENUM CROWN, we denote by $\mathsf{Sol}(G, x)$ the set of solutions. In the same way, we define $\mathsf{Sol}(G)$ for ENUM SMALL CROWN.

▶ Remark 6. Every instance of ENUM CROWN has a solution that takes the head and any x vertices in the crown. If M is an H-saturating matching between H and C, then every solution S of ENUM SMALL CROWN and every $e \in M$ satisfy $|S \cap V(e)| = 1$, where $V(e)$ denotes the endpoints of e.

Our kernelization algorithm consists in the exhaustive application of Rules VC.1 and VC.2, the latter of which can be applied in polynomial time due to Lemma 7.

▶ **Reduction Rule VC.1.** *Let (G, k) be the input instance of* ENUM VERTEX COVER. *If v is an isolated vertex of G, remove v from G and set $k \leftarrow k$.*

Proof of safeness and lifting algorithm of Rule VC.1. Let $G' = G \setminus \{v\}$. The forward direction is immediate; $S \in \mathsf{Sol}(G, k)$ if and only if $S \setminus \{v\} \in \mathsf{Sol}(G', k)$. For the converse direction, it is also immediate that $S' \in \mathsf{Sol}(G', k)$ if and only if $S' \in \mathsf{Sol}(G, k)$; however, observe that $(S' \cup \{v\}) \in \mathsf{Sol}(G, k)$ if and only if $|S'| < k$ as v is isolated, and we must also output $(S' \cup \{v\})$ if possible. ◀

As pointed out by Chlebík and Chlebíkova [15], the Nemhauser-Trotter [38] decomposition based on the half-integral relaxation of VERTEX COVER is a crown decomposition.

▶ **Lemma 7** ([38] and [15]). *Let G be a graph without isolated vertices and with at least $2k + 1$ vertices. Then, there is a polynomial-time algorithm that either:*

- *decides that no half-integral vertex cover of weight at most k exists; or*
- *finds a crown decomposition of G of width at most k.*

▶ **Reduction Rule VC.2** (Crown reduction). *Let (G, k) be an instance of* ENUM VERTEX COVER *with (C, H, B) a crown decomposition of G of width at most k. Set $G' \leftarrow G \setminus (H \cup C)$, $k' \leftarrow k - |H|$, and return (G', k').*

▶ **Lemma 8.** *If* ENUM CROWN *has a polynomial-delay enumeration algorithm $\mathcal{A}$, then Rule VC.2 is a reduction rule with an extension-only lifting algorithm.*

Proof. Let us show how to construct the lifting algorithm $\mathcal{R}_b$ using $\mathcal{A}$. Given a solution S' of the reduced instance (G', k'), let $x = k' - |S'|$ (where $x \geq 0$) be the *slack* of the solution S', and $F = N(B \setminus S') \cap H$ be the set of *forced* vertices in H, i.e. vertices that are adjacent to some vertex of B not picked in S'. Let $H' = H \setminus F$. For any $\ell \in [0, \min(|C|, x)]$, define $x_\ell = |H'| + \ell$, and observe that $(G[H' \cup C], x_\ell)$ is a valid instance of ENUM CROWN as $x_\ell \leq |C|$. For any $\ell \in [0, \min(|C|, x)]$, $\mathcal{R}_b$ outputs all solutions of the form $S' \cup F \cup S_\ell$, where $S_\ell \in \mathsf{Sol}(G[H' \cup C], x_\ell)$ are enumerated by $\mathcal{A}$. This completes the description of $\mathcal{R}_b$, so let us now prove that it adheres to the requirements of Definition 4. The first item is satisfied as it is well known that the crown reduction is safe for the decision version of the problem, in the sense that (G, k) is a yes-instance if and only if (G', k') is. Let us now check the second item, more precisely:

- For every $S' \in \mathsf{Sol}(G', k')$, the set $\mathcal{R}_b(G, G', k, k', S')$ is non-empty, and can be enumerated with polynomial-delay; and
- $\{\mathcal{R}_b(G, G', k, k', S') \mid S' \in \mathsf{Sol}(G', k')\}$ is a partition of $\mathsf{Sol}(G, k)$.

Let us start with the first property. Notice that there exists at least one $\ell \in [0, \min(|C|, x)]$, and as for any such ℓ, tuple $((H', C), x_\ell)$ is an instance of ENUM CROWN (which always has a solution by Remark 6), we get that $\mathcal{R}_b(G, G', k, k', S') \neq \emptyset$. Moreover, $\mathcal{R}_b(G, G', k, k', S')$ runs with polynomial-delay, as for each ℓ, $\mathcal{A}$ enumerates $\mathsf{Sol}(G[H' \cup C], x_\ell)$ with polynomial-delay, and there is no additional verification required in $\mathcal{R}_b$ to avoid duplicate solutions of the form $S' \cup F \cup S_\ell$ generated by different values of ℓ, as for any ℓ these solutions have size exactly $|S'| + |F| + x_\ell$.

Let us now consider the second property. Observe first that for any $S' \in \mathsf{Sol}(G', k')$, any $S \in \mathcal{R}_b(G, G', k, k', S')$ is indeed a solution of (G, k) as $V(G)$ can be partitioned into $V_1 = V(G') \cup F$ and $V_2 = H' \cup C$, and $S = S' \cup F \cup S_\ell$ where $S' \cup F$ is a vertex cover of $G[V_1]$, S_ℓ is a vertex cover of $G[V_2]$, and there is no edge between $V_1 \setminus S$ and V_2. Let us now prove that $\mathsf{Sol}(G, k) \subseteq \bigcup_{S' \in \mathsf{Sol}(G', k')} \mathcal{R}_b(G, G', k, k', S')$. Let $S \in \mathsf{Sol}(G, k)$ and $S' = S \setminus (H \cup C)$. As there is a matching of size $|H|$ between the head and the crown, $|S \cap (H \cup C)| \geq |H|$ and thus $S' \in \mathsf{Sol}(G', k')$. Now, let us prove that $S \in \mathcal{R}_b(G, G', k, k', S')$. Observe first that as S necessarily contains F, S is partitioned into $S = S' \cup F \cup (S \cap (H' \cup C))$, where $S \cap (H' \cup C)$ is a vertex cover of $G[H' \cup C]$. Let ℓ_0 be such that $|S \cap (H' \cup C)| = |H'| + \ell_0$. It remains to prove that $\ell_0 \in [0, \min(|C|, x)]$, as this implies that $(S \cap (H' \cup C)) \in \mathsf{Sol}(G[H' \cup C], x_{\ell_0})$, and thus that $S \in \mathcal{R}_b(G, G', k, k', S')$. First, we have $\ell_0 \geq 0$ as $(C, H', \emptyset)$ remains a crown decomposition, and thus there is a matching saturating H' in $G[H' \cup C]$. Let $x = k' - |S'|$ where $x \geq 0$. Moreover, $|S| = |S'| + |F| + |(S \cap (H' \cup C))| = k' - x + |F| + |H'| + \ell_0 = k - x + \ell_0 \leq k$, implying that $\ell_0 \leq x$. As $\ell_0 \leq |C|$ by definition, we get the claimed inequality. Finally, observe that $S \cap V(G') = S'$, so it follows that $\mathcal{R}_b$ is an extension-only lifting algorithm, which implies that, for any two different solutions S'_1, S'_2 of $\mathsf{Sol}(G', k')$, the sets $\mathcal{R}_b(G, G', k, k', S'_1)$, and $\mathcal{R}_b(G, G', k, k', S'_2)$ are disjoint. ◀

With this conditional result, we are now able to state a conditional kernelization lemma.

▶ **Lemma 9** ($\star$). *If* ENUM CROWN *has a polynomial-delay enumeration algorithm, then* ENUM VERTEX COVER *admits a PDE kernel (with extension-only lifting algorithm) with at most* $2k$ *vertices.*

The remainder of this section is dedicated to proving that ENUM CROWN admits a polynomial-delay enumeration algorithm. In what follows, given a matching M, and a subset $T \subseteq V(M)$ such that for any $e \in M$, $|V(e) \cap T| \leq 1$, we define $M_V(T) := \{v' \mid vv' \in M \text{ and } v \in T\}$ and call it the *image* of T by M. We need the following procedure and its associated technical lemma to obtain our lifting algorithm. Intuitively, in a small crowned graph G, exactly one endpoint of each edge of an H-saturating matching M can be in a vertex cover; $\mathsf{propagate_{dir}}(G, M, X_0)$ decides which vertices must be included and which must be excluded from a solution if we fix X_0 to be part of the it.

▶ **Definition 10** ($\mathsf{propagate_{dir}}(G, M, X_0)$). *Given a small crowned graph $G = (H \cup C, E)$ (meaning with $|H| = |C|$) and $X_0 \subseteq H$, the procedure $\mathsf{propagate_{dir}}(G, M, X_0)$ outputs two sets $(F, \bar{F})$ as follows (see Figure 1). Let $\vec{G} = (A \cup B, \vec{E})$ be an oriented bipartite graph with $A = H$, $B = C$, where:*

- *for any edge e between A and B with $e \in M$, we add its oriented version from A to B.*
- *for any edge e between A and B with $e \notin M$, we add its oriented version from B to A.*

Let $Z \subseteq A \cup B$ be the vertices that can be reached from X_0 in $\vec{G}$, including X_0 itself. Then, $\mathsf{propagate_{dir}}(G, M, X_0)$ outputs $(F, \bar{F})$ where $F = Z \cap A$ and $\bar{F} = Z \cap B$.

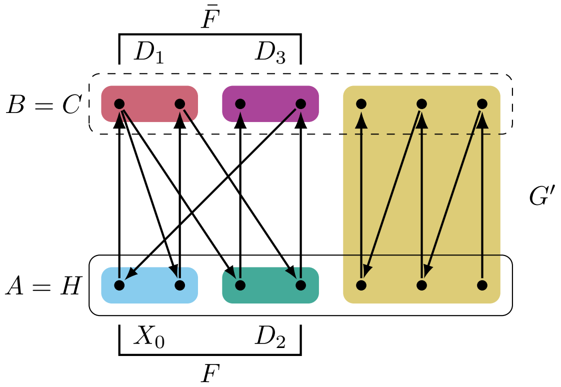


Figure 1 Example of an application of $\mathsf{propagate}_{\mathsf{dir}}(G, M, X_0)$ that stopped at distance three from X_0, with the graph depicted corresponding to $\vec{G}$. Edges of M correspond to the vertical arcs, while sloped arcs correspond to all other arcs of $\vec{G}$. The light-blue-shaded vertices correspond to X_0, the red-shaded to D_1 (the vertices at distance one from X_0), the teal-shaded to D_2, the purple-shaded to D_3, and the yellow-shaded vertices correspond to the graph $G' = G[(H \setminus F) \cup (C \setminus \bar{F})]$.

▶ **Lemma 11.** *Let G be a small crowned graph, M a perfect matching of G between H and C, $X_0 \subseteq H$, and $(F, \bar{F}) = \mathsf{propagate}_{\mathsf{dir}}(G, M, X_0)$. Then:*

1. *$\bar{F} = M_V(F)$ (where $F \subseteq H$ and $\bar{F} \subseteq C$) and for any solution $S \in \mathsf{Sol}(G)$ such that $X_0 \subseteq S$, S contains F and avoids $\bar{F}$.*
2. *For any solution $S \in \mathsf{Sol}(G)$ such that $X_0 \subseteq S$, if we define $H' = H \setminus F$ and $C' = C \setminus \bar{F}$),then $G' = G[H' \cup C']$ is a small crowned graph and $S \cap (C' \cup H') \in \mathsf{Sol}(G')$.*
3. *For any solution $S' \in \mathsf{Sol}(G')$, $F \cup S'$ is a solution of G (that contains X_0).*

Proof. Let $\vec{G} = (A \cup B, \vec{E})$ be the oriented bipartite graph defined in Definition 10.

Proof of Item 1. Let $S \in \mathsf{Sol}(G)$ such that $X_0 \subseteq S$. For any $i \geq 0$, let D_i be the vertices of (A, B) at distance i from X_0 in $\vec{G}$ (e.g. $D_0 = X_0$, $D_1 = M_V(X_0)$, see Figure 1). Let us show by induction that for any i, if i is even, then $D_i \subseteq A \cap S$, and if i is odd then $D_i \subseteq B \setminus S$, and $D_i = M_V(D_{i-1})$. Observe that the statement holds for $i = 0$. Let us now assume that it holds for $i - 1$ and consider i. If i is odd, observe first that $D_i = M_V(D_{i-1})$ by the definition of $\vec{G}$, and, by Remark 6, for any edge $v, v' \in M$ with $v \in D_{i-1}$ and $v' \in D_i$, $v' \notin S$. If i is even, as $D_{i-1} \cap S = \emptyset$ and S is a vertex cover of G, then $D_i \subseteq S$, and, as $\vec{G}$ is bipartite, we also get that $D_i \subseteq A$. Now, as $F = \bigcup_{i \equiv 0 \mod 2} D_i$ and $\bar{F} = \bigcup_{i \equiv 1 \mod 2} D_i$, we get $\bar{F} = M_V(F)$, and Item 1 follows.

Proof of Item 2. Observe first that G' is indeed a crowned graph as $\bar{F} = M_V(F)$. Now, as $S \in \mathsf{Sol}(G)$ we get $|S| = |H|$; by Remark 6 we get $|S \cap (F \cup \bar{F})| = |F|$, leading to $|S \cap (C' \cup H')| = |H'|$.

Proof of Item 3. Let $S' \in \mathsf{Sol}(G')$, and let us consider $S = F \cup S'$. Observe that $H \cup C$ is partitioned into $F \cup \bar{F}$ and $H' \cup C'$, that F is a vertex cover of $G[F \cup \bar{F}]$ (as $\bar{F} = M_V(F)$), and S' is a vertex cover of $G[H' \cup C']$. By definition of F and $\bar{F}$, there is no edge between $\bar{F}$ and $H' \cup C'$, implying that S is a vertex cover of $G[H \cup C]$. Since $|S| = |H|$, we get that S is a solution of G (that contains X_0 by definition of F). ◀

▶ **Lemma 12.** *If* Enum Small Crown *admits a polynomial-delay algorithm, then* Enum Crown *also admits such an algorithm.*

Proof. Let us define a polynomial-delay enumeration algorithm $\mathcal{A}$ for Enum Crown. Let (G, x) be an instance of Enum Crown with $V(G) = H \cup C$, and let M be a matching from H to C that saturates H. Given $T \subseteq V(G)$, we define $M(T) = \{e \in M \mid V(e) \subseteq T\}$, i.e., the set of edges of M entirely in T. For any $S \in \mathsf{Sol}(G, x)$, the *signature of* S is composed of two sets, $C_1(S) = V(M(S)) \cap C$ and $C_2(S) = (C \setminus V(M)) \cap S)$. Observe that

- $|C_1(S)| = |M(S)|$, where $|C_1(S)| \leq x$ (as $|C_1(S)| + |H| \leq |S| = |H| + x$);
- $x - (|C| - |M|) \leq |C_1(S)|$ (as if $x - (|C| - |M|) > |C_1(S)|$ then as $C_2(S) \subseteq (C \setminus V(M))$, we would have $|S| = |H| + |C_1(S)| + |C_2(S)| < |H| + x$), and $|C_2(S)| = x - |C_1(S)|$;
- By defining $\bar{X}_1(S) = (C \setminus V(M)) \setminus C_2(S)$, we have $S \cap \bar{X}_1(S) = \emptyset$.

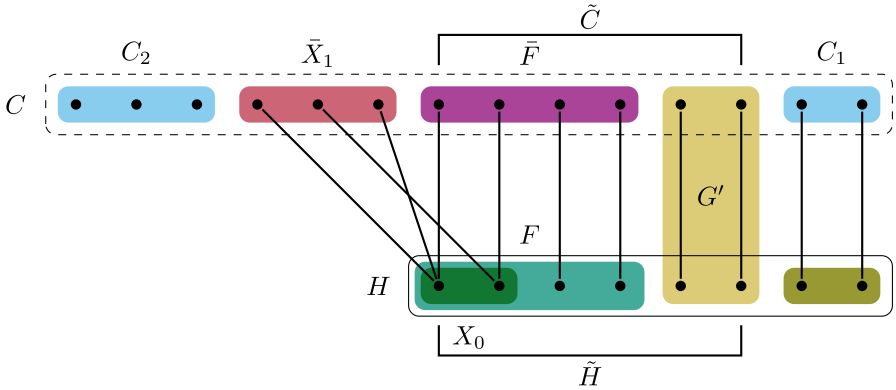


Figure 2 Example of the structures involved in the algorithm $\mathsf{propagate}_{\mathsf{large}}$, used in the proof of Lemma 12. The five light-blue-shaded vertices correspond to the inputs C_1, C_2 of $\mathsf{propagate}_{\mathsf{large}}$; the two olive-shaded vertices are forced to be in the solution by our choice of C_1. $\bar{X}_1$, i.e., the set of vertices of $C \setminus V(M)$ that should be forbidden in a solution to G, is represented by the three red-shaded vertices; the dark-green shaded vertices are forced to be in every solution that avoids $\bar{X}_1$, and, together with $\tilde{G} = G[\tilde{H} \cup \tilde{C}]$, composes the input to the call to $\mathsf{propagate}_{\mathsf{dir}}$ used in $\mathsf{propagate}_{\mathsf{large}}$; the four teal-shaded and four purple-shaded vertices are the outputs $F, \bar{F}$ of $\mathsf{propagate}_{\mathsf{dir}}$, and, together with $\bar{X}_1$, make up the output of $\mathsf{propagate}_{\mathsf{large}}$. Finally, the four yellow-shaded vertices represent G', the instance of Enum Small Crown to which $\mathcal{A}_s$ will be applied.

Our next step is guessing which edges of the H-saturating matching M will have both endpoints in the solution S; in the next propagation procedure, this is represented by set C_1, i.e. which vertices of $C \cap V(M)$ must be accompanied by their image in S. We also guess which vertices of C that *are not* matched to H will be included in a solution; this is represented by C_2. Finally, the choice of C_1 and C_2 implies that the vertices $\bar{X} \subseteq C \setminus (V(M) \cup C_2)$ are incident to yet uncovered edges. Formally, for any $d \in [\max(0, x - (|C| - |V(M))|), x]$, any set $C_1 \subseteq V(M) \cap C$ of size d, and any set $C_2 \subseteq C \setminus V(M)$ of size $x - d$, we define (see Figure 2) $\mathsf{propagate}_{\mathsf{large}}(C_1, C_2)$ that computes $(F, \bar{F}, \bar{X}_1)$ where $\bar{X}_1 = (C \setminus V(M)) \setminus C_2$, and $(F, \bar{F}) = \mathsf{propagate}_{\mathsf{dir}}(\tilde{G}, X_0)$ with $X_0 = N(\bar{X}_1)$, and $\tilde{G} = G[\tilde{H} \cup \tilde{C}]$, where $\tilde{H} = H \setminus M_V(C_1)$, and $\tilde{C} = M_V(\tilde{H})$.

▷ Claim 13. For any $S \in \mathsf{Sol}(G, x)$ of signature (C_1, C_2),

1. S contains F and avoids $\bar{F} \cup \bar{X}_1$;
2. By defining $H' = \tilde{H} \setminus F$ and $C' = \tilde{C} \setminus \bar{F}$, $G' = G[H' \cup C']$ is a small crowned graph and $(S \cap (C' \cup H')) \in \mathsf{Sol}(G')$.

Proof. Let $S \in \mathsf{Sol}(G, x)$ have signature (C_1, C_2). By definition of $C_2(S)$, we get that $S \cap \bar{X_1} = \emptyset$, implying also that $X_0 \subseteq S$. Moreover, by definition of $C_1(S)$, no edge $e \in M$ with $V(e) \subseteq \tilde{H} \cup \tilde{C}$ has both its endpoints in S. Thus, if we define $\tilde{S} = S \cap (\tilde{H} \cup \tilde{C})$, we get that $\tilde{S} \in \mathsf{Sol}(\tilde{G})$ and that $X_0 \subseteq \tilde{S}$. By Lemma 11, we obtain the two claimed properties. $\lhd$

We are now ready to define $\mathcal{A}$. To this end, let $\mathcal{A}_s$ be the polynomial-delay algorithm for Enum Small Crown. For any $d \in [\max(0, x - (|C| - |V(M)|)), x]$, any set $C_1 \subseteq V(M) \cap C$ of size d, and any set $C_2 \subseteq C \setminus V(M)$ of size $x - d$, $\mathcal{A}$ runs $\mathsf{propagate}_{\mathsf{large}}(C_1, C_2)$ that computes $(F, \bar{F}, \bar{X_1})$, and defines G', H', and C' as in Item 2 of Claim 13. Note that G' may be the empty graph. Then, $\mathcal{A}$ outputs all solutions of the form $C_1 \cup M(C_1) \cup C_2 \cup F \cup S'$, where $S' \in \mathsf{Sol}(G')$ are enumerated by $\mathcal{A}_s$. Observe that we can indeed call $\mathcal{A}_s$ on G' as according to Item 2, G' is a small crowned graph.

Let us first prove that there is no repetition. Observe that for each d, C_1, C_2 considered by $\mathcal{A}$, all enumerated solutions of the form $C_1 \cup M(C_1) \cup C_2 \cup F \cup S'$ (where $S' \in \mathsf{Sol}(G')$) have a signature (C_1, C_2). Now, it is sufficient to observe that two solutions of (G, x) with different signatures are necessarily different, which implies that there is no repetition.

Let us now prove that all enumerated elements are in $\mathsf{Sol}(G, x)$. Let $S = C_1 \cup M(C_1) \cup C_2 \cup F \cup S'$, where $S' \in \mathsf{Sol}(G')$ exists (as $S' = S \cap V(G')$ and G' is an induced subgraph of G) and is enumerated by $\mathcal{A}$. By Item 3 of Lemma 11, $(S' \cup F) \in \mathsf{Sol}(\tilde{G})$ and contains X_0. Now, observe that $H \cup C$ is partitioned into $V_1 = C_1 \cup M(C_1) \cup C_2 \cup \bar{X_1}$, and $V_2 = \tilde{H} \cup \tilde{C}$. Notice that $S \cap V_i$ is a vertex cover of $G[V_i]$ for $i \in \{1, 2\}$. Moreover, as $V_1 \setminus S = \bar{X_1}$, the only edges between V_1 and V_2 not covered by $S \cap V_1$ are edges between $\bar{X_1}$ and $N(\bar{X_1}) = X_0$, but as $S \cap V_2 \supseteq X_0$, S is indeed a vertex cover of G. Finally, $|S| = 2|C_1| + |C_2| + |F| + |H'| = 2|C_1| + |C_2| + |\tilde{H}| = |H| + |C_1| + |C_2| = |H| + x$, and so we have $S \in \mathsf{Sol}(G, x)$.

Towards proving that all elements of $\mathsf{Sol}(G, x)$ are enumerated, let $S \in \mathsf{Sol}(G, x)$, and $(C_1(S), C_2(S))$ be its signature. By definition of $\mathcal{A}$, there exists C_1 and C_2 considered by $\mathcal{A}$ such that $C_i = C_i(S)$ for $i \in \{1, 2\}$. Let $(F, \bar{F}, \bar{X_1}) = \mathsf{propagate}_{\mathsf{large}}(C_1, C_2)$. By Item 1, $S \supseteq F$, and $S \cap (\bar{F} \cup \bar{X_1}) = \emptyset$, which implies that $S = C_1 \cup M(C_1) \cup C_2 \cup F \cup S'$. As $S \cap (C' \cup H') = S'$, Item 2 implies that $S' \in \mathsf{Sol}(G')$, and so S' is enumerated by $\mathcal{A}_s$.

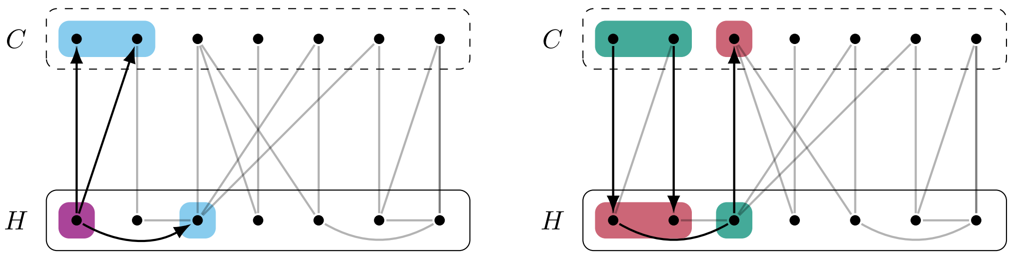
It remains to prove that there is only polynomial-time between two consecutive outputs of $\mathcal{A}$. Notice first that $\mathsf{propagate}_{\mathsf{large}}$ runs in polynomial time. Then, for each fixed C_1, C_2 considered by $\mathcal{A}$, as $\mathcal{A}_s$ runs in polynomial-delay, there is at most polynomial-time between two consecutive solutions having the same signature (C_1, C_2). Moreover, and as for any choice of C_1, C_2 considered by $\mathcal{A}$, we know that $\mathsf{Sol}(G') \neq \emptyset$ (by Remark 6), there is also a polynomial-delay between two consecutive solutions with different signatures. ◀

The proof of Lemma 16 requires a more involved variant of the propagation algorithm, where the goal is to avoid a prescribed vertex v.

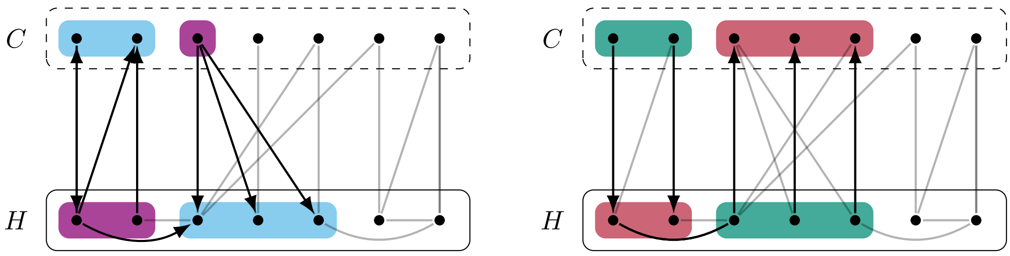
▶ **Definition 14** ($\mathsf{propagate}_{\mathsf{all}}(G, v)$). *Given a small crowned graph $G = (H \cup C, E)$ (i.e., $|H| = |C|$), and $v \in H$, the procedure* $\mathsf{propagate}_{\mathsf{all}}(G, v)$ *either* fails, *or outputs two sets $(F, \bar{F})$ that, intuitively, are forced to be included or avoided in a solution that avoids v and, upon removal, will yield a small crowned graph. We recursively define them as follows. Let $\bar{F_0} = \{v\}$. Observe that $N(\bar{F_0})$ intersects C (because of the perfect matching M between H and C, and may intersect H). Now, for any $i \geq 1$ (see Figure 3):*

- *to compute F_i: if there exists $e \in M$ such that $V(e) \subseteq N(\bar{F}_{i-1})$, we fail, and otherwise we define $F_i = N(\bar{F}_{i-1})$.*
- *to compute $\bar{F_i}$: if $M(F_i)$ is not an independent set, we fail, and otherwise we define $\bar{F_i} = M(F_i)$; note that $\bar{F}_{i-1} \subseteq \bar{F_i}$.*

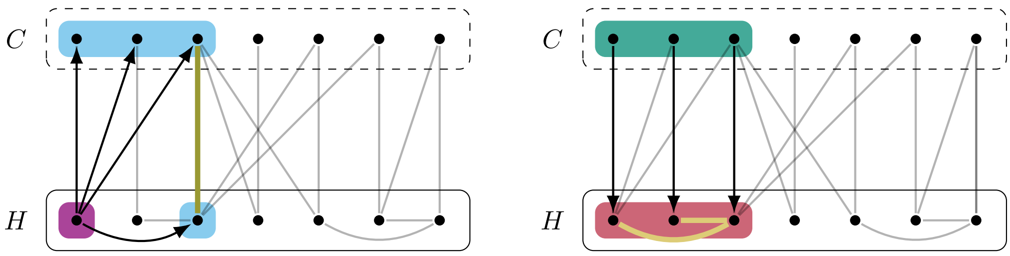
We continue this process until it fails or until $F_i = F_{i-1}$ and $\bar{F_i} = \bar{F}_{i-1}$. If there is no failure, $\mathsf{propagate}_{\mathsf{all}}(G, v)$ *returns $(F, \bar{F}) = (F_i, \bar{F_i})$.*



(a) Computing F_1 (cyan) from $\bar{F}_0$ (purple). **(b)** Computing $\bar{F}_1$ (red) from F_1 (teal).



(c) Computing F_2 (cyan) from $\bar{F}_1$ (purple). **(d)** Computing $\bar{F}_2$ (red) from F_2 (teal).



(e) Failing to compute F_1 (cyan) from $\bar{F}_0$ (purple). **(f)** Failing to compute $\bar{F}_1$ (red) from F_1 (teal).

Figure 3 Examples of executions of algorithm $\mathsf{propagate_{all}}$. Vertical edges correspond to M. We reduce the width of the edges of G not involved in the corresponding operation for clarity. An arc (x, y) indicates that vertex y was included in the set currently being built by virtue of the membership of x in the root set. The edges that cause the failures in the computation of F_i or $\bar{F}_i$ are thickened and colored differently.

▶ **Lemma 15.** *Let $G = (H \cup C, E)$ be a small crowned graph, and $v \in H$.*

1. *If $\mathsf{propagate_{all}}(G, v)$ fails, then there is no $S \in \mathsf{Sol}(H, C)$ such that $v \notin S$.*
2. *Otherwise, if $\mathsf{propagate_{all}}(G, v)$ returns $(F, \bar{F})$ (where $\bar{F} = M_V(F)$), then for any $S \in \mathsf{Sol}(G)$ such that $v \notin S$, $F \subseteq S$ and $S \cap \bar{F} = \emptyset$; moreover, if we define $H' = H \setminus (F \cup \bar{F})$ and $C' = C \setminus (F \cup \bar{F})$, then $G' = G[H' \cup C']$ is a crowned graph with $|C'| = |H'|$, and $S \cap (C' \cup H') \in \mathsf{Sol}(G')$.*
3. *For any $S' \in \mathsf{Sol}(G')$, $F \cup S'$ is a solution of G that avoids v.*

Proof. Suppose that there is some $S \in \mathsf{Sol}(G)$ where $v \notin S$. Let us prove by induction that for any $i \geq 1$, if there is a failure when computing F_i or $\bar{F}_i$, then S cannot exist, and otherwise $F_i \subseteq S$ and $S \cap \bar{F}_i = \emptyset$.

Assume there was no failure, and consider the step where we try to compute F_i. By induction, $S \cap \bar{F}_{i-1} = \emptyset$, so we have that $N(\bar{F}_{i-1}) \subseteq S$ and, by construction, $F_i = N(\bar{F}_{i-1})$. By Remark 6, if there is an edge $e \in M$ where $V(e) \subseteq F_i$, then no such solution exists. Let us now consider the case where we try to compute $\bar{F}_i$. Again by induction, we have $F_{i-1} \subseteq S$, and by Remark 6 we know that no edge of M can have both endpoints in S, implying that we must have $S \cap M_V(F_i) = \emptyset$. Thus, if $G[M_V(F_i)]$ is not an independent set, then no such solution exists.

The correctness of Item 1 now directly follows from the induction. The proof of Item 2 is also immediate: if $\mathsf{propagate_{all}}(G, v)$ returns $(F, \bar{F})$, then $\bar{F} = M_V(F)$ follows from the definition, and for $S \in \mathsf{Sol}(G)$ such that $v \notin S$, $F \subseteq S$ and $S \cap \bar{F} = \emptyset$ follows from the induction, and the fact that $(S \cap (C' \cup H')) \in \mathsf{Sol}(G')$ is true by Remark 6.

To prove Item 3, let $S' \in \mathsf{Sol}(G')$, and let $S = F \cup S'$. Notice that $H \cup C$ is partitioned into $F \cup \bar{F}$ and $H' \cup C'$, that F is a vertex cover of $G[F \cup \bar{F}]$ (as $\bar{F} = M_V(F)$), and that S' is a vertex cover of $G[H' \cup C']$. Now, let i_0 be the index where $\mathsf{propagate_{all}}$ stops, meaning that $F_{i_0} = F_{i_0-1}$. This implies that $N(\bar{F}_{i_0}) \subseteq F_{i_0-1}$, and thus that there is no edge between $\bar{F}_{i_0} = \bar{F}$ and $H' \cup C'$, implying that S is a vertex cover of $G[H \cup C]$. As $|S| = |H|$, we get that S is a solution of G that avoids v by definition of F. ◀

With the propagation algorithm in hand, Lemma 16 can be proved by constructing a branching algorithm where we either add a vertex v of H or do not add it to the solution. The key properties we use are that: (i) there is always one solution that uses v (e.g. taking the entirety of H), and (ii) we can decide in polynomial time if it is possible to avoid v in a solution. Consequently, we recursively call our branching algorithm only if we know that there is some solution to be found in this branch.

▶ **Lemma 16.** *There is a polynomial-delay algorithm for* Enum Small Crown.

Proof. Let $G = (H \cup C, E)$ be a small crowned graph and let us define an algorithm $\mathcal{A}_s$ that enumerates all vertex covers of G of size $|H|$. If $V(G) = \emptyset$, then $\mathcal{A}_s$ returns $\emptyset$, so now let $v \in H$ be an arbitrary vertex. Let $(F_v, \bar{F}_v) = \mathsf{propagate_{dir}}(G, \{v\})$, $H'_v = H \setminus F_v$, $C'_v = C \setminus \bar{F}_v$, and $G'_v = G[H'_v \cup C'_v]$. Algorithm $\mathcal{A}_s$ outputs first all solutions of the form $F_v \cup S'$, where S' are enumerated by a recursive call $\mathcal{A}_s(G'_v)$. Then, $\mathcal{A}_s$ calls $\mathsf{propagate_{all}}(G, v)$. If it fails then $\mathcal{A}_s$ stops, otherwise let $(F_{\bar{v}}, \bar{F}_{\bar{v}})$ be the output of $\mathsf{propagate_{all}}(G, \bar{v})$, $H'_{\bar{v}} = H \setminus F_{\bar{v}}$, $C'_{\bar{v}} = C \setminus \bar{F}_{\bar{v}}$, and $G'_{\bar{v}} = G[H'_{\bar{v}} \cup C'_{\bar{v}}]$. Algorithm $\mathcal{A}_s$ outputs then all solutions of the form $F_{\bar{v}} \cup S'$, where S' are enumerated by a recursive call $\mathcal{A}_S(G'_{\bar{v}})$. This concludes the definition of $\mathcal{A}_s$.

First, observe that the recursive calls are made on small crowned graphs, as they were obtained by removing sets $(F, \bar{F})$ from $V(G)$ where $\bar{F} = M_V(F)$. The fact that there is no repetition is immediate by induction, as in particular all solutions of $F_v \cup S'$, where $S' \in \mathsf{Sol}(G'_v)$, contain v, and all solutions of the form $F_{\bar{v}} \cup S''$, where $S'' \in \mathsf{Sol}(G'_{\bar{v}})$, do not contain v.

Let us prove by induction that all enumerated elements are in $\mathsf{Sol}(G)$. We first consider enumerated sets of the form $F_v \cup S'$, where S' are enumerated by $\mathcal{A}_s(G'_v)$. By induction, we get that $S' \in \mathsf{Sol}(G'_v)$, and by Item 3 of Lemma 11, we get that $F_v \cup S' \in \mathsf{Sol}(G)$. Now, for enumerated sets of the form $F_{\bar{v}} \cup S''$, where S'' are enumerated by $\mathcal{A}_s(G'_{\bar{v}})$, the induction hypothesis implies that $S' \in \mathsf{Sol}(G'_{\bar{v}})$, and again by Item 3 of Lemma 15, we have that $F_{\bar{v}} \cup S'' \in \mathsf{Sol}(G)$.

Let us now prove by induction that all elements of $\mathsf{Sol}(G)$ are enumerated. Let $S \in \mathsf{Sol}(G)$. If $v \in S$, then by Lemma 11, $S = F_v \cup S'$, where $S' \in \mathsf{Sol}(G'_v)$, and by induction S' will be enumerated by $\mathcal{A}_s(G'_v)$. If $v \notin S$, then by Lemma 15, as $\mathsf{propagate_{all}}(G, v)$ cannot fail (as S exists), $S = F_{\bar{v}} \cup S'$, where $S' \in \mathsf{Sol}(H'_{\bar{v}}, C'_{\bar{v}})$, and by induction S' will be enumerated by $\mathcal{A}_s(H'_{\bar{v}}, C'_{\bar{v}})$.

Finally, there is at most polynomial time between any two solutions, as both $\mathsf{propagate_{dir}}$ and $\mathsf{propagate_{all}}$ run in polynomial time, and because for any recursive call made on an input $G' \in \{G'_v, G'_{\bar{v}}\}$, we know according to Remark 6 that $\mathsf{Sol}(G') \neq \emptyset$. Moreover, the branching tree has height bounded by $|V(G)|$ and every node either: is a leaf, and so falls in the base case where $V(G) = \emptyset$; has exactly one child, corresponding to $\mathcal{A}_s(G'_v)$, for which a solution always exists; or has two children where $\mathcal{A}_s(G'_v)$ has the same guarantee as in the previous case, and $\mathcal{A}_s(G'_{\bar{v}})$ is called if and only if we know that $\mathsf{Sol}(G'_{\bar{v}}) \neq \emptyset$ by Lemma 15. Consequently, every leaf of the branching tree corresponds to a solution of G and the time taken between visiting two of them is bounded by a polynomial in $|V(G)| + |E(G)|$, guaranteeing the desired polynomial delay. ◀

The following theorem is now a direct consequence of Lemmas 9, 12, and 16.

▶ **Theorem 1.** Enum Vertex Cover *admits a PDE kernel with at most* $2k$ *vertices when parameterized by the maximum desired size of a solution* k*.*

4 Feedback Vertex Set

Throughout this section, we take (G, k) to be our input instance to Enum Feedback Vertex Set, let $X \subseteq V(G)$ be a 2-approximation using the algorithm given in [2], $\mathsf{fvs}(G)$ denote the size of a minimum feedback vertex set, and $\Delta(G)$ its maximum degree. Thomassé's [39] strategy can be split into four phases: (i) apply reduction rules to bound the maximum and minimum degrees of the instance; (ii) prove that $|V(G)| \in \mathcal{O}(\Delta(G) \cdot \mathsf{fvs}(G))$ in graphs of minimum degree three; (iii) identify if there is a vertex v in X that belong to too many vertex-disjoint cycles in $(G \setminus X) \cup \{v\}$; and (iv) once no such v exists, pick a high-degree vertex u, compute a 2-expansion (i.e. a crown decomposition where each head vertex is matched to two unique crown vertices) in an auxiliary graph, and remove all edges from u to the vertices identified by the expansion. As exhaustively applying (iv) implies a bound on the maximum degree, (i) and (ii) will yield the desired kernel. We follow a similar kernelization strategy, but use a substantially different set of rules beyond the basic ones; our presentation is closer to that of Fomin et al. [28, Chapter 5]. We present a generalization of point (i) in Lemma 20: $|V(G)| \in \mathcal{O}(\Delta(G) \cdot \mathsf{fvs}(G))$ holds even in some of graphs of minimum degree two.

Due to space limitations, all proofs of safeness and associated lifting algorithms of the reductions rules presented in this section have been omitted, and can be found in the full version available online.

We begin by bounding the minimum degree of G; like other works dealing with multigraphs, we define $\deg_G(v)$ as the number of edges incident to a vertex v in G, and omit the subscript when G is clear from the context. We say that $uv \in E(G)$ is a *double-edge* if the multiplicity of uv in $E(G)$, denoted by $\mathsf{mul}(uv)$, is two. Note that a double-edge is equivalent to a cycle of length two in G or, equivalently, that at least one of its endpoints must be present in *every* solution of the instance. Our first rule is only used to simplify the graph.

▶ **Reduction Rule FVS.1** ($\star$)**.** *Perform the following modifications:*

i. *If there is an edge uv of G of multiplicity at least three, set its multiplicity to two.*
ii. *If there is some $v \in V(G)$ of degree at most one, remove v.*
iii. *If $u, v \in V(G)$ have $k + 2$ common neighbors, add edges uv until* $\mathsf{mul}(uv) = 2$.
iv. *If there is some vertex v incident to a loop or to $k + 1$ distinct double-edges, remove v and set $k \leftarrow k - 1$.*

Let us briefly discuss double-edges and their implications to a Feedback Vertex Set instance. Typically, double-edges can be divided into two groups: *real* edges, where we detect that one of the endpoints in fact belongs to every solution, and *virtual* edges, that are used to simplify the instance by showing that at least one solution (if any exists) contains one of the endpoints. The key reduction rule of Thomassé [39], which yields the required linear degree bound, makes heavy usage of virtual edges; unfortunately, we do not know how to manage the (potentially exponentially) many solutions that are forbidden in the reduced instance by the introduction of these virtual edges, or an alternative rule to linearly bound the maximum degree. More generally, virtual edges seem to make the lifting process much more complicated. As such we make a special effort to avoid them. In particular, this implies that the naive reduction rule that replaces a degree-two vertex in a triangle with a parallel edge between its neighbors is out of the question. We remark that, while this rule does have an associated lifting procedure, we hope that maintaining only real edges can be leveraged in the future to develop a smaller PDE kernel for Enum Feedback Vertex Set.

In the following rules, we always assume that we are also given an arbitrary but fixed total ordering σ of the vertices of G. This is needed to avoid repeatedly outputting a solution during lifting.

▶ **Reduction Rule FVS.2** ($\star$). *If three vertices $a, x, b \in V(G)$ form an induced path with $N(x) = \{a, b\}$ and $\sigma(a) < \sigma(b)$, then contract x into a.*

We note that the following rule does introduce a virtual double-edge, but we remark that, if applied, it will inevitably trigger Rule FVS.4, and so we do not violate our own constraints.

▶ **Reduction Rule FVS.3** ($\star$). *If $u, x, y \in V(G)$ form a triangle with $N[x] = N[y] = \{u, x, y\}$, and $\sigma(x) < \sigma(y)$, then contract y into x, i.e., make $\mathsf{mul}(ux) = 2$.*

▶ **Reduction Rule FVS.4** ($\star$). *If $u \in V(G)$ shares double-edges with $t \geq 1$ different vertices $\varepsilon_u = \{v_1, \ldots, v_t\}$ with $\deg(v_i) = 2$ for every $i \in [t]$, remove u, every v_i, and set $k \leftarrow k - 1$.*

▶ **Reduction Rule FVS.5** ($\star$). *If there is a pair of vertices $a, b \in V(G)$ and a set of degree-two vertices $\varepsilon_{ab} = \{x_1, \ldots, x_t\}$ with $N(x_i) = \{a, b\}$ for every $i \in [t]$ and $N(b) = \{a\} \cup \varepsilon_{ab}$, then remove $N[b]$ and set $k \leftarrow k - 1$.*

▶ **Lemma 17** ($\star$). *Let G be a graph and Δ its maximum degree. If none of the Rules FVS.1–FVS.5 are applicable, then $\mathsf{fvs}(G) \geq |V(G)|/(3\Delta - 3)$.*

With Lemma 17 at hand, we are tasked with bounding the maximum degree of G. Formally, a *v-flower of order t* is a family of t cycles that pairwise intersect only at vertex v. Rule FVS.6 and Lemma 18 are taken straight from [39]; we omit the proof of the latter for brevity, while we give a proof for the former as we must discuss how to lift the solutions of the reduced instance back to the input instance.

▶ **Reduction Rule FVS.6** ($\star$). *If there is a vertex $v \in V(G)$ and an v-flower of order at least $k + 1$, remove v and set $k \leftarrow k - 1$.*

▶ **Lemma 18** ([39]). *Let G a multigraph, X a feedback vertex set of G of size at most $2k$, and $x \in V(G)$. In polynomial time, it is possible to accomplish exactly one of the following:*
1. *Decide if (G, k) has $\mathsf{Sol}(G, k) = \emptyset$, i.e., the corresponding decision problem is a* no*-instance;*
2. *Find a v-flower of order $k + 1$; or*
3. *Find a set feedback vertex set H_x such that $X \setminus \{x\} \subseteq H_x \subseteq V(G) \setminus \{x\}$ of size at most $3k$.*

What is left now is to show how to leverage the third property of Lemma 18. To do so, suppose that Rule FVS.6 is not applicable, take $v \in V(G)$ with $\deg(v) > 3k(k+1) + 5k$, let H_v be as in Lemma 18, and let $\mathcal{C}$ denote the set of connected components of $G \setminus (H_v \cup \{v\})$. Note that, as $X \subseteq H_v \cup \{v\}$, each $C_i \in \mathcal{C}$ is a tree.

▶ **Observation 19** (⋆). *There exists $\mathcal{D}_v \subseteq \mathcal{C}$ with $|\mathcal{D}_v| > 3k(k+1)$ such that, for every $D_i \in \mathcal{D}_v$: v is adjacent to one vertex of D_i with a non-double-edge, and D_i has at least one vertex adjacent to a vertex of H_v.*

At this point, we essentially have the same setup as Thomassé [39]. Instead of leveraging q-expansions, however, we take a simpler approach, which does not yield a linear bound but allows us to obtain a polynomial PDE kernel; we remark that it is still interesting to explore if these generalizations of crown decompositions can be leveraged to reduce the quadratic bound on the maximum degree that we obtain. We are first going to slightly increase the degree of v, before decreasing it, as follows.

▶ **Reduction Rule FVS.7** (⋆). *Let $\mathcal{D}_v, H_v$ be as above, and G_v be the bipartite graph with bipartition $(\{d_i \mid D_i \in \mathcal{D}_v\}, \{h_u \mid u \in H_v\})$, and $d_ih_u \in E(G_v)$ if and only if $N_G(u) \cap D_i \neq \emptyset$. For every $h_u \in V(G_v)$ of degree at least $k+2$, add a double-edge uv to G.*

We remark that the double-edges introduced by Rule FVS.7 are not virtual: there are $k+2$ cycles with pairwise intersection equal to $\{u, v\}$.

▶ **Reduction Rule FVS.8** (⋆). *Let H_v^2 be the vertices of H_v with a double-edge to v. If there exists $D_i \in \mathcal{D}_v$ with $N_G(D_i) \setminus \{v\} \subseteq H_v^2$, remove vw from G, where $w \in D_i \cap N_G(v)$.*

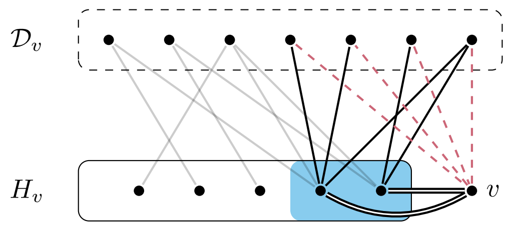


Figure 4 Example of repeated applications of Rule FVS.8; where each $D_i \in \mathcal{D}_v$ is contracted into a single vertex for ease of presentation. The two cyan-shaded vertices correspond to H_v^2. The double-edges incident to v are real double-edges, the seven light-gray edges are incident to vertices of $\mathcal{D}_v$ not affected by the rule, the four red dashed edges are the ones removed by the reduction rule, and the five solid edges are incident to vertices of $\mathcal{D}_v$ affected by the rule.

Rule FVS.8 is our key rule to reduce the maximum degree of the graph; we present an example of repeated applications of it in Figure 4. We are now ready to bound the maximum degree of G by a function of k; along with Lemma 17, this will imply a bound on the overall size of G.

▶ **Lemma 20.** *If Rules FVS.7 and FVS.8 are not applicable, then G has maximum degree at most $3k(k+1) + 5k$.*

Proof. Suppose that there is some $v \in V(G)$ incident to more than $3k(k+1) + 5k$ edges. By Observation 19 and Rule FVS.6, the set $\mathcal{D}_v$ of connected components of $G \setminus (H_v \cup \{v\})$ that v has a neighbor in, but no double-edges to, has size greater than $3k(k+1)$; moreover, every $D_i \in \mathcal{D}_v$ has a neighbor in H_v. Now, take some $u \in H_v$ for which $\mathsf{mul}(uv) < 2$; as Rule FVS.7 is not applicable, we know that $N_G(u)$ hits at most $k+1$ components of $\mathcal{D}_v$. As such, taking H_v^2 as in Rule FVS.8, we know that $N_G(H_v \setminus H_v^2)$ hits at most $|H_v \setminus H_v^2| \cdot (k+1)$ components of $\mathcal{D}_v$. Note that, if $H_v^2 = \emptyset$ and $|\mathcal{D}_v| > |H_v| \cdot (k+1)$, then there is some $h \in H_v$ adjacent to vertices in at least $k+2$ components of $\mathcal{D}_v$ and Rule FVS.7 is applicable, so it must be the case that $H_v^2 \neq \emptyset$. As such, since at most $(|H_v| - 1)(k+1)$ components of $\mathcal{D}_v$ have a vertex adjacent to a vertex in $H_v \setminus H_v^2$, it follows that there is at least one component of $\mathcal{D}_v$ whose neighborhood is contained in $H_v^2 \cup \{v\}$, which is enough to trigger Rule FVS.8. ◀

▶ **Lemma 21.** *There is a polynomial-time algorithm that, given an instance* (G, k) *of* Enum Feedback Vertex Set*, either answers that* $\mathsf{Sol}(G, k) = \emptyset$ *or outputs an equivalent instance* (G', k') *where* $|V(G)| \leq 3k^3 + 8k^2$*.*

Proof. The algorithm simply consists of the exhaustive application of Rules FVS.1–FVS.8, which also guarantee that (G, k) and (G', k') are equivalent. By Lemma 20, we have that the maximum degree Δ of G' is at most $3k(k + 1) + 5k$. By Lemma 17, we have that $\mathsf{fvs}(G') \geq |V(G')|/(3\Delta - 3)$. As such, if $k'(3k'(k' + 1) + 5k') < |V(G')|$, then we know that no feedback vertex set of size k' exists in G', and so $\mathsf{Sol}(G', k') = \mathsf{Sol}(G, k) = \emptyset$. Otherwise, $|V(G')| \leq 3k'^3 + 3k'^2 + 5k'^2 = 3k'^3 + 8k'^2$ and we are done. ◀

▶ **Lemma 22** (⋆)**.** *There is a polynomial-delay algorithm that, given* (G, k)*,* (G', k')*, and* $S' \in \mathsf{Sol}(G', k')$*, returns a non-empty set* $\mathsf{Lift}(S') \subseteq \mathsf{Sol}(G, k)$*. Moreover,* $\mathsf{Lift}(G', k) := \{\mathsf{Lift}(S') \mid S' \in \mathsf{Sol}(G', k')\}$ *is a partition of* $\mathsf{Sol}(G, k)$*.*

Finally, the proof of Theorem 2 follows immediately from Lemmas 21 and 22.

▶ **Theorem 2.** *There is a PDE kernel for* Enum Feedback Vertex Set *parameterized by the size of the solution* k *with* $\mathcal{O}(k^3)$ *vertices.*

5 Final remarks

In this paper, we have developed polynomial-sized PDE kernels for the natural parameterizations of enumeration variants of some key parameterized problems, namely Enum Vertex Cover and Enum Feedback Vertex Set. In this process, we showed how to leverage crown decompositions, an important technique initially developed for decision kernelization, to the enumeration setting. We hope that these results can further motivate the study of this new domain of parameterized complexity. Among concrete open problems, we are interested in the enumeration of minimal vertex covers and feedback vertex sets; we believe our techniques are applicable to these variants of the studied problems. A more challenging question is the existence of a quadratic kernel for Enum Feedback Vertex Set, which we do not see how to obtain using our approach; it seems like tools describing global structures of the graph, like q-expansions, are needed. Finally, we remark that lower bound theories for enumerative kernelization and enumerative parameterized complexity would be of extreme benefit to the area. Currently, lower bounds are derived almost solely via the related parameterized *decision* problems. It is unlikely, however, that this strategy is enough to (conditionally) classify parameterized enumeration problems into tractable and intractable and which admit polynomial PDE kernels and those that do not.

References

1 Faisal N. Abu-Khzam, Michael R. Fellows, Michael A. Langston, and W. Henry Suters. Crown structures for vertex cover kernelization. *Theory of Computing Systems*, 41(3):411–430, 2007. doi:10.1007/s00224-007-1328-0.

2 Vineet Bafna, Piotr Berman, and Toshihiro Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal on Discrete Mathematics*, 12(3):289–297, 1999. doi:10.1137/S0895480196305124.

3 Guillaume Bagan, Arnaud Durand, and Etienne Grandjean. On acyclic conjunctive queries and constant delay enumeration. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic*, pages 208–222, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi:10.1007/978-3-540-74915-8_18.

4 Max Bannach, Zacharias Heinrich, Rüdiger Reischuk, and Till Tantau. Dynamic kernels for hitting sets and set packing. *Algorithmica*, 84(11):3459–3488, 2022. doi:10.1007/s00453-022-00986-0.

5 Valentin Bartier, Oscar Defrain, and Fionn Mc Inerney. Hypergraph dualization with FPT-delay parameterized by the degeneracy and dimension. In *Combinatorial Algorithms: 35th International Workshop, IWOCA 2024, Ischia, Italy, July 1–3, 2024, Proceedings*, pages 111–125, Berlin, Heidelberg, 2024. Springer-Verlag. doi:10.1007/978-3-031-63021-7_9.

6 Stéphane Bessy, Marin Bougeret, Dimitrios M. Thilikos, and Sebastian Wiederrecht. *Kernelization for Graph Packing Problems via Rainbow Matching*, pages 3654–3663. Society for Industrial and Applied Mathematics, 2023. doi:10.1137/1.9781611977554.ch139.

7 Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (Meta) kernelization. *J. ACM*, 63(5), November 2016. doi:10.1145/2973749.

8 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM Journal on Discrete Mathematics*, 28(1):277–305, 2014. doi:10.1137/120880240.

9 Hans L. Bodlaender and Thomas C. van Dijk. A cubic kernel for feedback vertex set and loop cutset. *Theor. Comp. Sys.*, 46(3):566–597, April 2010. doi:10.1007/S00224-009-9234-2.

10 J.A. Bondy and U.S.R Murty. *Graph Theory*. Springer Publishing Company, Incorporated, 1st edition, 2008.

11 Marin Bougeret, Bart M. P. Jansen, and Ignasi Sau. Bridge-depth characterizes which minor-closed structural parameterizations of vertex cover admit a polynomial kernel. *SIAM J. Discret. Math.*, 36(4):2737–2773, 2022. doi:10.1137/21M1400766.

12 Kevin Burrage, Vladimir Estivill-Castro, Michael Fellows, Michael Langston, Shev Mac, and Frances Rosamond. The undirected feedback vertex set problem has a poly(k) kernel. In *Proceedings of the Second International Conference on Parameterized and Exact Computation*, IWPEC'06, pages 192–202, Berlin, Heidelberg, 2006. Springer-Verlag. doi:10.1007/11847250_18.

13 Jonathan F. Buss and Judy Goldsmith. Nondeterminism within P. In Christian Choffrut and Matthias Jantzen, editors, *STACS 91*, pages 348–359, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg. doi:10.1007/BFB0020811.

14 Guilherme C. M. Gomes, Emanuel Juliano, Gabriel Martins, and Vinicius F. dos Santos. Matching (Multi)Cut: Algorithms, Complexity, and Enumeration. In Édouard Bonnet and Paweł Rzążewski, editors, *19th International Symposium on Parameterized and Exact Computation (IPEC 2024)*, volume 321 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:15, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.IPEC.2024.25.

15 Miroslav Chlebík and Janka Chlebíková. Crown reductions for the minimum weighted vertex cover problem. *Discrete Applied Mathematics*, 156(3):292–312, 2008. Combinatorial Optimization 2004. doi:10.1016/j.dam.2007.03.026.

16 Nadia Creignou, Markus Kröll, Reinhard Pichler, Sebastian Skritek, and Heribert Vollmer. A complexity theory for hard enumeration problems. *Discrete Applied Mathematics*, 268:191–209, 2019. doi:10.1016/j.dam.2019.02.025.

17 Nadia Creignou, Arne Meier, Julian-Steffen Müller, Johannes Schmidt, and Heribert Vollmer. Paradigms for parameterized enumeration. *Theory of Computing Systems*, 60(4):737–758, May 2017. doi:10.1007/s00224-016-9702-4.

18 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

19 Peter Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. *Theoretical Computer Science*, 351(3):337–350, 2006. Parameterized and Exact Computation. doi:10.1016/j.tcs.2005.10.004.

20 Peter Damaschke. Fixed-parameter enumerability of cluster editing and related problems. *Theory of Computing Systems*, 46(2):261–283, 2010. doi:10.1007/s00224-008-9130-1.

21 Holger Dell and Dániel Marx. Kernelization of packing problems. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 68–81, USA, 2012. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611973099.6.

22 Rodney G Downey and Michael R Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

23 Arnaud Durand, Nicole Schweikardt, and Luc Segoufin. Enumerating answers to first-order queries over databases of low degree. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '14, pages 121–131, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2594538.2594539.

24 Henning Fernau. On parameterized enumeration. In Oscar H. Ibarra and Louxin Zhang, editors, *Computing and Combinatorics*, pages 564–573, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. doi:10.1007/3-540-45655-4_60.

25 Fedor V. Fomin, Tien-Nam Le, Daniel Lokshtanov, Saket Saurabh, Stéphan Thomassé, and Meirav Zehavi. Subquadratic kernels for implicit 3-hitting set and 3-set packing problems. *ACM Trans. Algorithms*, 15(1), January 2019. doi:10.1145/3293466.

26 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM Journal on Discrete Mathematics*, 30(1):383–410, 2016. doi:10.1137/140997889.

27 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Dimitrios M. Thilikos. Bidimensionality and kernels. *SIAM Journal on Computing*, 49(6):1397–1422, 2020. doi:10.1137/16M1080264.

28 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: Theory of Parameterized Preprocessing.* Cambridge University Press, 2019. doi:10.1017/9781107415157.

29 Valentin Garnero, Christophe Paul, Ignasi Sau, and Dimitrios M. Thilikos. Explicit linear kernels via dynamic programming. *SIAM Journal on Discrete Mathematics*, 29(4):1864–1894, 2015. doi:10.1137/140968975.

30 Petr A. Golovach, Christian Komusiewicz, Dieter Kratsch, and Van B. Le. Refined notions of parameterized enumeration kernels with applications to matching cut enumeration. *Journal of Computer and System Sciences*, 123:76–102, 2022. doi:10.1016/j.jcss.2021.07.005.

31 David S. Johnson, Mihalis Yannakakis, and Christos H. Papadimitriou. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988. doi:10.1016/0020-0190(88)90065-8.

32 Eun Jung Kim, Alexander Langer, Christophe Paul, Felix Reidl, Peter Rossmanith, Ignasi Sau, and Somnath Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. *ACM Trans. Algorithms*, 12(2), December 2015. doi:10.1145/2797140.

33 Christian Komusiewicz, Diptapriyo Majumdar, and Frank Sommer. Polynomial-size enumeration kernelizations for long path enumeration, 2025. doi:10.48550/arXiv.2502.21164.

34 Mithilesh Kumar and Daniel Lokshtanov. A 2lk Kernel for l-Component Order Connectivity. In Jiong Guo and Danny Hermelin, editors, *11th International Symposium on Parameterized and Exact Computation (IPEC 2016)*, volume 63 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.IPEC.2016.20.

35 A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960. URL: http://www.jstor.org/stable/1910129.

36 Diptapriyo Majumdar. Enumeration kernels of polynomial size for cuts of bounded degree, 2024. arXiv:2308.01286.

37 Kitty Meeks. Randomised enumeration of small witnesses using a decision oracle. *Algorithmica*, 81(2):519–540, 2019. doi:10.1007/s00453-018-0404-y.

38 G. L. Nemhauser and L. E. Trotter. Vertex packings: Structural properties and algorithms. *Mathematical Programming*, 8(1):232–248, 1975. doi:10.1007/BF01580444.

39 Stéphan Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algorithms*, 6(2), April 2010. doi:10.1145/1721837.1721848.